(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 029 267 B1

(12) EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention of the grant of the patent:
27.03.2002 Bulletin 2002/13

(21) Application number: 98959455.1

(22) Date of filing: 13.11.1998

(51) Int Cl.^{7}: G06F 9/00

(86) International application number:
PCT/US98/24378

(87) International publication number:
WO 99/26133 (27.05.1999 Gazette 1999/21)

(54) METHOD FOR MAINTAINING THE SYNCHRONIZED EXECUTION IN FAULT RESILIENT/FAULT TOLERANT COMPUTER SYSTEMS

VERFAHREN ZUR ERHALTUNG VON SYNCHRONISIERTER AUSFÜHRUNG BEI FEHLER-BETRIEBSSICHEREN/ FEHLERTOLERANTEN RECHNERSYSTEMEN

SYSTEME INFORMATIQUE PRESENTANT UNE RESILIENCE ET UNE TOLERANCE FACE AUX DEFAILLANCES

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE

(30) Priority: 14.11.1997 US 65790 P

(43) Date of publication of application:
23.08.2000 Bulletin 2000/34

(73) Proprietor: Marathon Technologies Corporation
Boxboro, MA 01719 (US)

(72) Inventors:
- BISSETT, Thomas, D.
  Kingston, RI 02881 (US)
- LEVEILLE, Paul, A.
  Grafton, MA 01519 (US)
- MUENCH, Erik
  Groveland, MA 01834 (US)

(74) Representative: Jones, David Colin et al
Withers & Rogers,
Goldings House
2 Hays Lane
London SE1 2HW (GB)

(56) References cited:
EP-A- 0 286 856
EP-A- 0 372 580
EP-A- 0 411 805
WO-A-93/09494
US-A- 5 226 152
US-A- 5 600 784

EP 1 029 267 B1

Printed by Jouve, 75001 PARIS (FR)

Description

TECHNICAL FIELD

[0001] The invention relates to maintaining synchronized execution by processors in fault resilient/fault tolerant computer systems.

BACKGROUND

[0002] Computer systems that are capable of surviving hardware failures or other faults generally fall into three categories: fault resilient, fault tolerant, and disaster tolerant.

[0003] Fault resilient computer systems can continue to function, often in a reduced capacity, in the presence of hardware failures. These systems operate in either an availability mode or an integrity mode, but not both. A system is "available" when a hardware failure does not cause unacceptable delays in user access, which means that a system operating in an availability mode is configured to remain online, if possible, when faced with a hardware error. A system has data integrity when a hardware failure causes no data loss or corruption, which means that a system operating in an integrity mode is configured to avoid data loss or corruption, even if the system must go offline to do so.

[0004] Fault tolerant systems stress both availability and integrity. A fault tolerant system remains available and retains data integrity when faced with a single hardware failure, and, under some circumstances, when faced with multiple hardware failures.

[0005] Disaster tolerant systems go beyond fault tolerant systems. In general, disaster tolerant systems require that loss of a computing site due to a natural or man-made disaster will not interrupt system availability or corrupt or lose data.

[0006] All three cases require an alternative component that continues to function in the presence of the failure of a component. Thus, redundancy of components is a fundamental prerequisite for a disaster tolerant, fault tolerant or fault resilient system that recovers from or masks failures. Redundancy can be provided through passive redundancy or active redundancy, each of which has different consequences.

[0007] A passively redundant system, such as a checkpoint-restart system, provides access to alternative components that are not associated with the current task and must be either activated or modified in some way to account for a failed component. The consequent transition may cause a significant interruption of service. Subsequent system performance also may be degraded. Examples of passively redundant systems include stand-by servers and clustered systems. The mechanism for handling a failure in a passively redundant system is to "fail-over", or switch control, to an alternative server. The current state of the failed application may be lost, and the application may need to be restarted in the other system. The fail-over and restart processes may cause some interruption or delay in service to the users. Despite any such delay, passively redundant systems such as stand-by servers and clusters provide "high availability" and do not deliver the continuous processing usually associated with "fault tolerance."

[0008] An actively redundant system, such as a replication system, provides an alternative processor that concurrently processes the same task and, in the presence of a failure, provides continuous service. The mechanism for handling failures is to compute through a failure on the remaining processor. Because at least two processors are looking at and manipulating the same data at the same time, the failure of any single component should be invisible both to the application and to the user.

[0009] The goal of a fault tolerant system is to produce correct results in a repeatable fashion. Repeatability ensures that operations may be resumed after a fault is detected. In a checkpoint-restart system, this entails rolling back to a previous checkpoint and replaying the inputs again from a journal file. In a replication system, repeatability results from simultaneous operation on multiple instances of a computer.

[0010] Many fault tolerant designs are known for single processor systems. There also are a few known fault tolerant, symmetric multi-processing ("SMP") systems. The extra complexity associated with providing fault tolerance in an SMP system causes problems for many traditional approaches to fault tolerance.

[0011] For a checkpoint-restart system, the checkpoint information is somewhat more complex, but the recovery algorithm remains basically the same. Repeatability can be loosely interpreted to permit the replay of system operation to occur differently than the original system operation. In other words, the allocation of workload between SMP processors on the replay does not have to follow the allocation that was being followed when the fault occurred. The order of the inputs must be preserved, but the relative timing of the inputs to each other and to the instruction streams running on the different processors does not need to be preserved.

[0012] Under this loose repeatability standard, a replay is valid as long as the results produced by the replay are proper for the sequence of inputs. An example is an airline reservation system with multiple customers (e.g., Mr. Smith and Ms. Jones) competing for the last seat. Due to input timing and processor scheduling, Ms. Jones gets the seat. However, before the result is posted, a fault occurs. On the replay, Mr. Smith gets the seat. Though producing a different result, the replay is valid since there is no cognizable problem associated with the change in result (i.e., Ms. Jones will never know she almost got the seat).

[0013] SMP adds considerable complexity to replication systems. Corresponding processors in corresponding systems must produce the same results at the same time. The input timing must be precisely preserved with respect to the multiple instruction streams. No difference between processor arbitration cycles is allowed, because such a difference can affect who gets what resource first. Making an SMP system with replication requires control of all aspects of the system that can affect the timing of input data and the arbitration between processors.

[0014] For these reasons, fault tolerant SMP systems generally are produced using the checkpoint-restart approach. In such systems, the application and operating system software must be specially designed to support checkpoints.

[0015] The document EP-A-0 286 856 teaches a fault tolerant symmetric multiprocessing system with strongly coupled compute elements.

SUMMARY

[0016] The invention, various aspects of which are described here below, is defined in detail in the appended claims 1 and 24.

[0017] In one general aspect, a fault tolerant/fault resilient computer system includes at least two compute elements connected to at least one controller. Each of the compute elements has clocks that operate asynchronously to clocks of the other compute elements. The compute elements operate in a first mode in which the compute elements each execute a first stream of instructions in emulated clock lockstep. Clock lockstep operation requires the compute elements to perform the same sequence of instructions in the same order, with each instruction being performed in the same clock cycle by each compute element. The compute elements also operate in a second mode in which the compute elements each execute a second stream of instructions in instruction lockstep. Instruction lockstep operation requires the compute elements to perform the same sequence of instructions in the same order, but does not require the compute elements to perform the instructions in the same clock cycle.

[0018] Implementations of the computer system may include one or more of the following features. For example, each compute element may be a multi-processor compute element, such as a symmetric multi-processor (SMP) compute element. Each compute element may be implemented using an industry standard motherboard. The system may be configured to deactivate all but one of the processors of each compute element when the compute elements are operating in the second mode.

[0019] The first stream of instructions may implement operating system and application software, while the second stream of instructions implements lockstep control software. The operating system and application software may be unmodified software configured for use with computer systems that are not fault tolerant.

[0020] Each compute element may include one or more processors, memory, and a connection to the controller. The compute elements may be configured so that refresh operations associated with the memory are synchronized with execution of operations by the processor. The system also may be configured to initiate DMA transfers to the memory when the compute elements are operating in the second mode and to execute the initiated DMA transfers when the compute elements are operating in the first mode.

[0021] The system may synchronize the compute elements by copying contents of the memory of a first compute element to the memory of a second compute element, and resetting the processors of the first and second compute elements in a way that does not affect the memories of the compute elements.

[0022] The compute elements may transition from the first mode of operation to the second mode of operation in response to an interrupt. For example, the interrupt may be a performance counter interrupt generated by the compute element after the occurrence of a fixed number of clock cycles, such as processor clock cycles or bus clock cycles. The interrupt also may be generated after the execution of a fixed number of instructions. When the compute elements are multi-processor compute elements having primary processors and one or more secondary processors, the primary processor may be configured to halt operation of the secondary processors in response to the interrupt.

[0023] Each compute element may generate an interrupt during the transition from the second mode of operation to the first mode of operation. This interrupt serves to align the processing by the compute element with a clocking structure of the compute element. Typically, the interrupt is synchronized with a clock having the lowest frequencies of the clocking structure.

[0024] The system may redirect I/O operations by the compute elements to the controller. The system also may include a second controller connected to the first controller and to the two compute elements. The first controller and a first compute element may be located in a first location and the second controller and a second compute element may be located in a second location, in which case the system also may include a communications link connecting the first controller to the second controller, the first controller to the second compute element, and the second controller to the first compute element. The first location may be spaced from the second location by more than 5 meters, by more than 100 meters, or even by a kilometer or more.

[0025] A benefit of creating a fault resilient/ fault tolerant SMP system using replication is that the system can run standard application and operating system software, such as the Windows NT operating system available from Microsoft Corporation. In addition, the system can do so using industry-standard processors and motherboards, such as motherboards based on Pentium series processors available from Intel Corporation.

[0026] Other features and advantages will be apparent from the following description, including the drawings, and from the claims.

DESCRIPTION OF DRAWINGS

[0027]

Figs. 1 and 2 are block diagrams of a fault resilient/ fault tolerant uni-processor computer system.
Fig. 3 is a block diagram of a fault resilient/fault tolerant multi-processor computer system.
Fig. 4 is a block diagram of a motherboard.
Fig. 5 is a flow chart of a procedure implemented by the system of Fig. 3.
Fig. 6 is a block diagram of a PCI interface.
Fig. 7 is a flow chart of a procedure implemented by the system of Fig. 3.
Fig. 8 is a block diagram of a system having two multi-processor compute elements and one I/O processor.
Figs. 9A and 9B are a flow chart of a procedure implemented by the system of Fig. 8.

DETAILED DESCRIPTION

[0028] The fault tolerant systems described below emulate fully-phase-locked operation of multiple instances of a compute element. This should be contrasted to prior systems that operated multiple instances of a compute element in instruction lockstep, such as the Endurance 4000 system available from Marathon Technologies Corporation of Boxboro, Massachusetts. Instruction lockstep operation occurs when multiple instances of a compute element perform the same sequence of instructions in the same order. Fully-phase-locked operation, which also may be referred to as clock lockstep operation, occurs when multiple instances of a compute element perform the same sequence of instructions in the same order, with each instruction being performed in the same clock cycle by each instance of the compute element.

[0029] In the Endurance 4000 system, the instances of a compute element operate in instruction stream lockstep. Each compute element executes the same sequence of instructions prior to producing an output. The time needed to execute the instruction stream varies due to the uncontrolled past history of each compute element. For example, caches, table lookahead buffers, branch prediction logic, speculative execution logic, and execution pipelines of the compute elements can have different initial values, which, even though the instruction streams being executed are the same, result in varying execution times.

[0030] Instruction lockstep operation may result in failures when the compute elements are SMP servers. In such a system, each compute element has multiple processors, each with its own instruction stream. The instruction streams are arbitrating for shared resources. This arbitration must be resolved identically in both compute elements for redundant operation. Instruction lockstep operation does not provide a tight enough control over the processors and the memory to guarantee the same arbitration resolution in both compute elements.

[0031] Clock lockstep operation may be achieved by using a common oscillator to provide clocks to all instances of the compute element. However, such an implementation may be unsuited for fault tolerant operation because it includes a single component, the common oscillator, the failure of which will cause failure of the entire system.

[0032] Emulated clock lockstep operation avoids the single point of failure and is achieved using the techniques described below. Emulated clock lockstep operation offers the considerable additional benefit of permitting the different instances of a compute element to be separated by distances of up to a kilometer or more.

[0033] An emulated-clock-lockstep, non-SMP, fault tolerant system is described below. This description is followed by description of a fault tolerant SMP system using replication and emulated-clock-lockstep operation. In both systems, the basic approach is to design a system in which multiple instances of a compute element are initialized into exactly the same state and then provided with exactly the same input stimuli from a synchronous I/O subsystem. This causes each instance to produce exactly the same result.

[0034] To progress a fault tolerant non-SMP (uniprocessor) implementation to a fault resilient/fault tolerant SMP implementation, each processor is replaced by several processors and an arbitration unit. Any time that a processor needs access to anything beyond its internal cache (e.g., memory or I/O), the processor uses the arbitration unit to arbitrate for the external bus that connects the processors together. Given that the arbitration units are finite state engines initialized to the same state, they will follow the same sequence of arbitrations as long as the processors are functioning correctly.

Uni-Processor (Non-SMP) System

[0035] Fig. 1 illustrates a fault tolerant, non-SMP system 100 that emulates clock lockstep operation. In general, all computer systems perform two basic operations: (1) manipulating and transforming data, and (2) moving the data to and from mass storage, networks, and other I/O devices. The system 100 divides these functions, both logically and physically, between two separate processors. For this purpose, each half of the system 100, called a tuple, includes a compute element ("CE") 105 and an I/O processor ("IOP") 110. The compute element 105 processes user application and operating system software. I/O requests generated by the compute element 105 are redirected to the I/O processor 110. This redirection is implemented at the device driver level. The I/O processor 110 provides I/O resources, including I/O processing, data storage, and network connectivity. The I/O processor 110 also controls synchronization of the compute elements.

[0036] The system 100 is fault tolerant in that it continues to operate transparently to its users in the presence of any single hardware failure. The system 100 emulates a traditional computing environment by partitioning it into two components. The compute element 105 handles all compute tasks for the operating system and any applications. The I/O processor 110 handles all I/O devices. Thus, the I/O processor handles all of the asynchronous activities associated with a computer, while the compute element handles all of the synchronous compute activities.

[0037] To provide the necessary redundancy for fault tolerance, the system 100 includes at least two compute elements 105 and at least two I/O processors 110. The two compute elements 105 operate in lockstep while the two I/O processors 110 are loosely coupled. The I/O processors 110 feed both compute elements 105 the exact same data at a controlled place in the instruction streams of the compute elements. The I/O processors verify that the compute elements generate the same I/O operations and produce the same output data at the same time. The I/O processors also cross check each other for proper completion of requested I/O activity.

[0038] The system 100 uses a software-based approach in a configuration based on inexpensive, industry standard processors. For example, the compute elements 105 and I/O processors 110 may be implemented using Pentium Pro processors available from Intel Corporation. The system may run unmodified, industry-standard operating system software, such as the Windows NT operating system available from Microsoft Corporation, as well as industry-standard applications software. This permits a fault tolerant system to be configured by combining off-the-shelf, Intel Pentium Pro-based servers from a variety of manufacturers, which results in a fault tolerant or disaster tolerant system with low acquisition and life cycle costs.

[0039] Each compute element 105 includes a processor 115, memory 120, and an interface card 125 (also referred to as a Marathon interface card, or MIC). The interface card 125 includes drivers for communicating with two I/O processors simultaneously, as well as comparison and test logic that assures results received from the two I/O processors are identical. In the fault tolerant system 100, the interface card 125 of each compute element 105 is connected by high speed links 130, such as fiber optic links, to interface cards 125 of the two I/O processors 110. The interface cards 125 may be implemented as PCI-based adapters.

[0040] Each I/O processor 110 includes a processor 115, memory 120, an interface card 125, and I/O adapters 135 for connection to I/O devices such as a hard drive 140 and a network 145. As noted above, the interface card 125 of each I/O processor 110 is connected by high speed links 130 to the interface cards 125 of the two compute elements 105. In addition, a high speed link 150, such as a private ethernet link, is provided between the two I/O processors 110.

[0041] All I/O task requests from the compute elements 105 are redirected to the I/O processors 110 for handling. The I/O processor 110 runs specialized software that handles all of the fault handling, disk mirroring, system management, and resynchronization tasks required by the system 100. By using a multitasking operating system, such as Windows NT, the I/O processor 110 may run other, non-fault tolerant applications. In general, a compute element may run Windows NT Server as an operating system while, depending on the way that the I/O processor is to be used, an I/O processor may run either Windows NT Server or Windows NT Workstation as an operating system.

[0042] The two compute elements 105 run lockstep control software, also referred to as quantum synchronization software, and execute the operating system and the applications in emulated clock lockstep. Disk mirroring takes place by duplicating writes on the disks 140 associated with each I/O processor. If one of the compute elements 105 should fail, the other compute element 105 keeps the system running with a pause of only a few milliseconds to remove the failed compute element 105 from the configuration. The failed compute element 105 then can be physically removed, repaired, reconnected, and turned on. The repaired compute element then is brought back automatically into the configuration by transferring the state of the running compute element to the repaired compute element over the high speed links and resynchronizing. The states of the operating system and applications are maintained through the few seconds it takes to resynchronize the two compute elements so as to minimize any impact on system users.

[0043] If an I/O processor 110 fails, the other I/O processor 110 continues to keep the system running. The failed I/O processor then can be physically removed, repaired and turned back on. Since the I/O processors are not running in lockstep, the repaired system may go through a full operating system reboot, and then may be resynchronized. After being resynchronized, the repaired I/O processor automatically rejoins the configuration and the mirrored disks are re-mirrored in background mode over the private connection 150 between the I/O processors. A failure of one of the mirrored disks is handled through the same process.

[0044] The connections to the network 145 also are fully redundant. Network connections from each I/O processor 110 are booted with the same address. Only one network connection is allowed to transmit messages, while both are allowed to receive messages. In this way, each network connection monitors the other through the private ethernet. Should either network connection fail, the I/O processors will detect the failure and the remaining connection will carry the load. The I/O processors notify the system manager in the event of a failure so that a repair can be initiated.

[0045] While Fig. 1 shows both connections on a single network segment, this is not a requirement. Each I/O processor's network connection may be on a different segment of the same network. The system also accommodates multiple networks, each with its own redundant connections. The extension of the system to disaster tolerance requires only that the connection between the tuples be optical fiber or a connection having compatible speed. With such connections, the tuples may be spaced by distances of a kilometer or more. Since the compute elements are synchronized over this distance, the failure of a component or a site will be transparent to the users.

[0046] Fig. 2 provides a summarized view of the system 100 of Fig. 1. The system includes redundant compute elements 105 ("CEs") and I/O processors 110 ("IOPs"). Each CE 105 is responsible for all computing and may be implemented using an industry standard motherboard. Each IOP 110 is responsible for access to I/O devices, and for system control. The IOPs run asynchronously of each other and verify that the CEs are performing the same operations in the same order. The IOPs also track each other's I/O completion to ensure that no I/O is lost.

[0047] The CEs generate the same outputs in the exact same sequence, and run in emulated clock lockstep, even though the CE clocks are asynchronous to each other. The CEs are initialized to the same state and are fed consistent inputs at exactly the same time. The CEs are periodically realigned using a self-generated interrupt that is related to the occurrence of a quantum of clock cycles (e.g., 100,000 clock cycles) and is referred to as a quantum interrupt ("QI"). By contrast, the prior Endurance 4000 system used QIs related to the completion of a quantum of instructions. All inputs to the CEs are delivered at either an output window or after the completion of an instruction quantum. Both of these points are guaranteed to occur at the same point in the instruction streams of the CEs. The approach employed by the Endurance 4000 system is described in U.S. Patent Nos. 5,600,784 and 5,615,403.

Multi-Processor (SMP) System

[0048] Fig. 3 illustrates a fault resilient/fault tolerant, symmetric multi-processing ("SMP") system 300. Each CE 305 of the system 300 includes a collection of processors 310 connected by a common processor bus 315 and an arbitration unit 320. The processors use the bus 315 and arbitration unit 320 to access a shared memory 325, and to access two IOPs 330 through an interface card 335 and high speed data links 340.

[0049] The IOPs 330 operate identically to the IOPs 110 of the system 100. Thus, the IOPs handle all I/O task requests from the processors 310 and run specialized software that handles all of the fault handling, disk mirroring, system management, and resynchronization tasks required by the system 300.

[0050] One processor 310 (identified as processor 310a) of each CE 305 serves as a primary processor and runs lockstep control software in addition to executing an operating system and applications in emulated clock lockstep with the other CE. The remaining processors in each CE 305 execute the operating system and applications in emulated clock lockstep with the other CE.

[0051] Referring to Fig. 4, a motherboard 400 for use in a CE 305 of the system 300 includes two or more processors 310. Each processor may operate at a clock speed of, for example, 300 MHz or 350 Mhz. The processors 310 are interconnected and connected to the arbitration unit 320 by the bus 315, which is also referred to as the processor bus or the front side bus ("FSB"). The FSB typically operates at a clock speed of 100 MHz. The arbitration unit 320 is commonly referred to as the North Bridge, since it serves as a bridge from the processor bus 315 to the memory 325 and to the PCI bus 705. The PCI bus 705 typically is a 32 bit bus operating at 33 MHz or a 64 bit bus operating at 66 Mhz. The interface card 335 is implemented as a PCI device connected to the PCI bus 705.

[0052] The PCI bus 705 is also connected to another component, which is commonly referred to as the South Bridge 710. The South Bridge includes an advanced peripheral interrupt controller ("APIC") 715 that provides interrupts to the processors 310 on an APIC bus 720. The processors 310 include their own APICs 725 that receive the interrupts. The APIC bus may be, for example, a 16.6 MHz bus.

[0053] The motherboard 700 may be implemented using an industry standard motherboard. In this case, the motherboard 700 also may include a number of components that, though standard on the motherboard, are not used by the system 300. These components include a video card 730 connected to the North Bridge 320 by an AGP bus 735 (or by the PCI bus); one or more SCSI controllers 740 connected to the PCI bus 705; one or more PCI devices 745 connected to the PCI bus 705; an IDE drive controller 750 connected to the South Bridge 710; an ISA (16 bit, 8 Mhz) or EISA (32 bit, 8 Mhz) bus 755 connected to the South Bridge 710; one or more ISA or EISA devices 760 connected to the bus 755; and a super I/O controller 765 connected to the bus 755 to provide keyboard, mouse, and floppy drive support, as well as parallel and serial ports. These components, if present, are not used by the CE 305.

[0054] Marathon's prior Endurance 4000 system provided a fault tolerant structure in which processors were kept in lockstep while disregarding time skew. In essence, the time difference between processors was not important, assuming asynchrony between processors did not affect instruction lockstep. Memory refresh and DMA interactions, which had no impact on the lockstep of the processors, did affect the timing asynchrony. Video processing had both a timing and an instruction component. Care was taken to ensure that video and quantum processing created neither instruction nor data di-

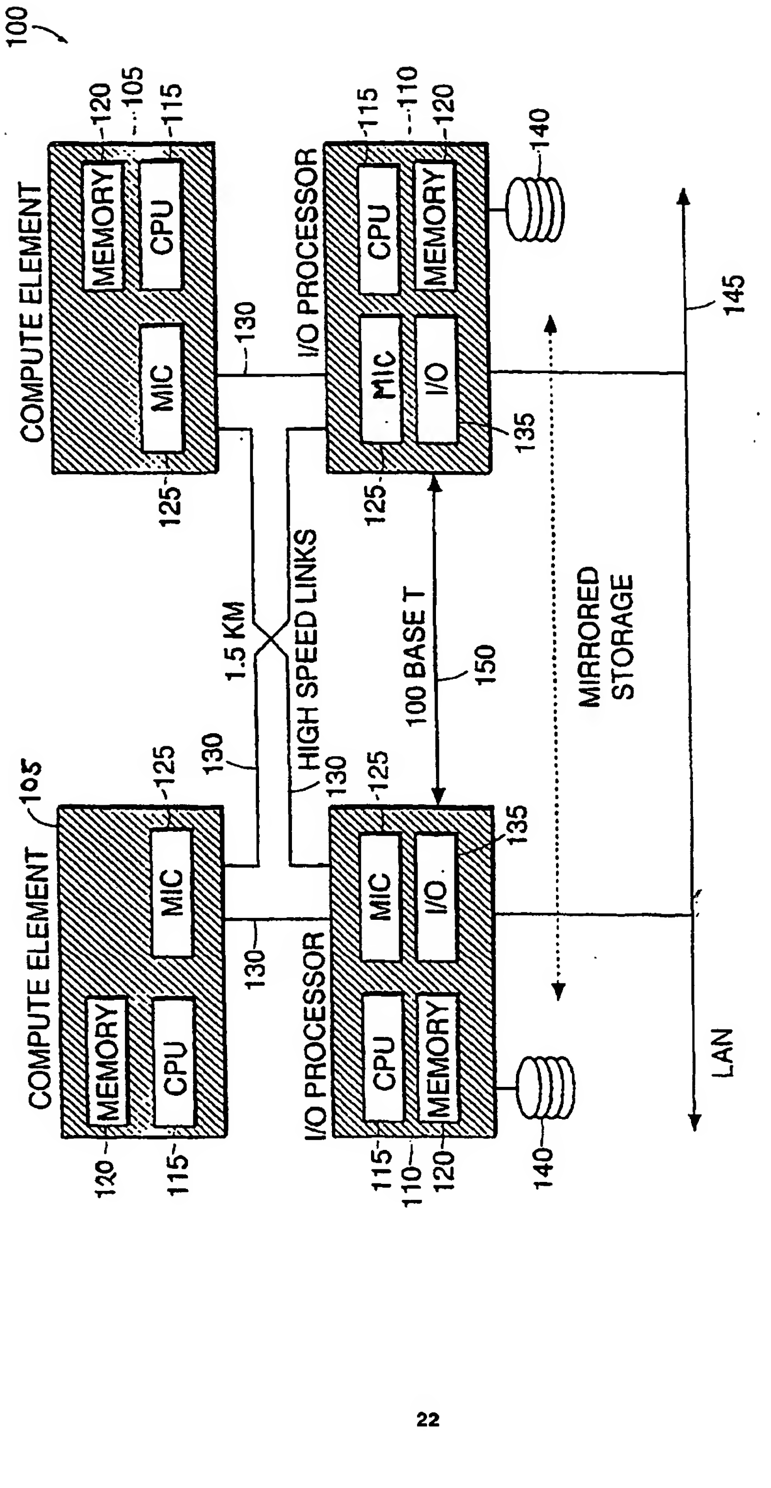
100
COMPUTE ELEMENT
MEMORY
CPU
MIC
120
105
115
125
130
I/O PROCESSOR
CPU
MEMORY
MIC
I/O
115
110
120
125
135
140
145
1.5 KM
HIGH SPEED LINKS
100 BASE T
150
MIRRORED STORAGE
LAN

FIG. 1

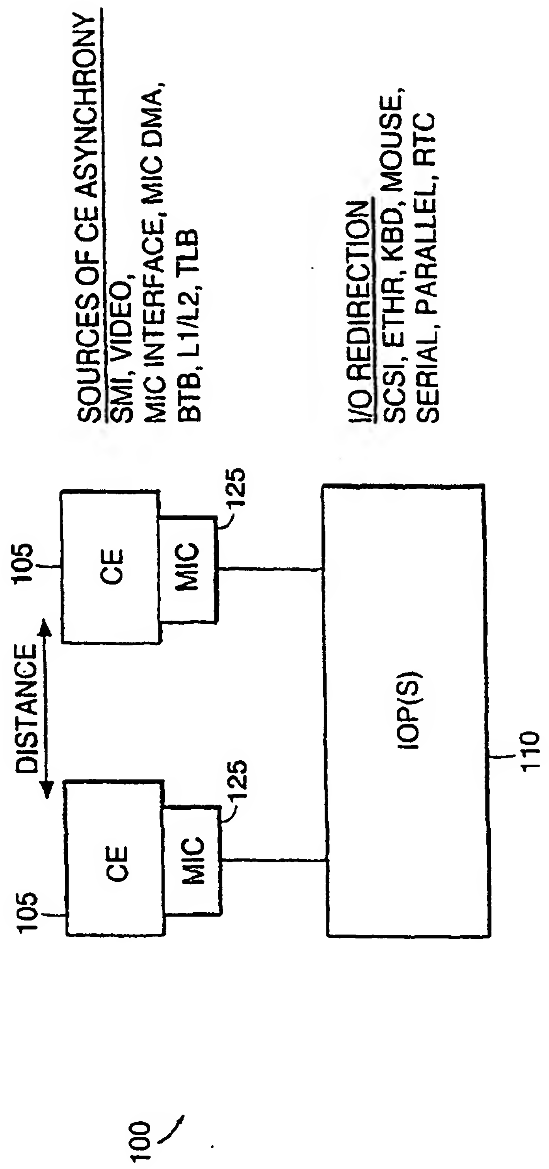
SOURCES OF CE ASYNCHRONY
SMI, VIDEO,
MIC INTERFACE, MIC DMA,
BTB, L1/L2, TLB
I/O REDIRECTION
SCSI, ETHR, KBD, MOUSE,
SERIAL, PARALLEL, RTC
105
CE
MIC
125
DISTANCE
105
CE
MIC
125
IOP(S)
110
100

FIG. 2

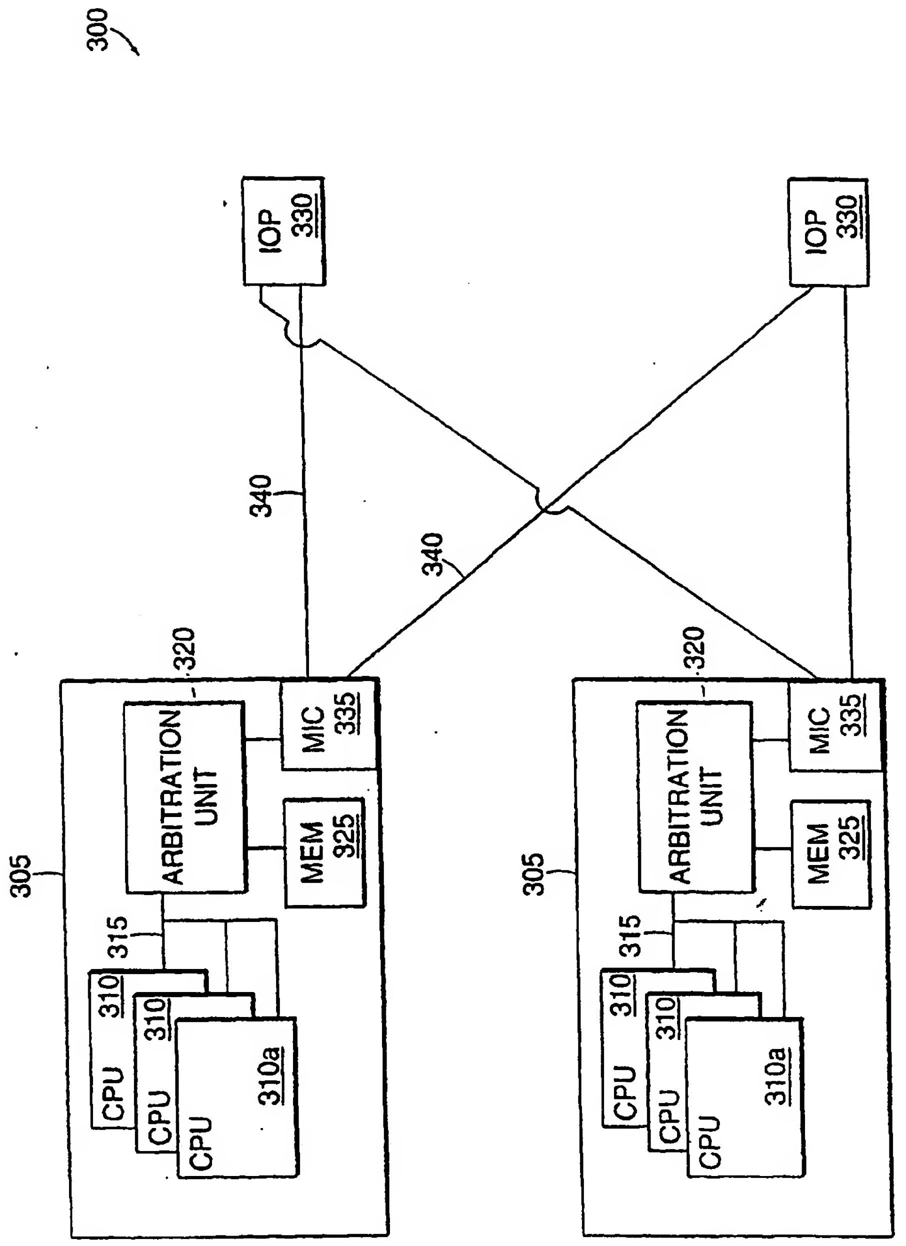
300
IOP 330
IOP 330
340
340
305
320
ARBITRATION UNIT
MIC 335
MEM 325
315
CPU 310
CPU 310
CPU 310a
305
320
ARBITRATION UNIT
MIC 335
MEM 325
315
CPU 310
CPU 310
CPU 310a

FIG. 3